Event-Based Communication in Distributed Q-Learning

Jarne Ornia, D.; Mazo, M.

DOI
10.1109/CDC51059.2022.9992660

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the IEEE 61st Conference on Decision and Control (CDC 2022)

2022 IEEE 61st Conference on Decision and Control (CDC)
December 6-9, 2022. Cancún, Mexico

Event-Based Communication in Distributed Q-Learning

Daniel Jarne Ornia
Delft Centre for Systems and Control
Delft University of Technology
Delft, The Netherlands
d.jarneornia@tudelft.nl

Manuel Mazo Jr.
Delft Centre for Systems and Control
Delft University of Technology
Delft, The Netherlands
m.mazo@tudelft.nl

2022 IEEE 61st Conference on Decision and Control (CDC) | 978-1-6654-6761-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/CDC51059.2022.9992660

***Abstract*—We present an approach to reduce the communication of information needed on a Distributed Q-Learning system inspired by Event Triggered Control (ETC) techniques. We consider a baseline scenario of a Distributed Q-Learning problem on a Markov Decision Process (MDP). Following an event-based approach, N agents sharing a value function explore the MDP and compute a trajectory-dependent triggering signal which they use distributedly to decide when to communicate information to a central learner in charge of computing updates on the action-value function. These decision functions form an Event Based distributed Q learning system (EBd-Q), and we derive convergence guarantees resulting from the reduction of communication. We then apply the proposed algorithm to a cooperative path planning problem, and show how the agents are able to learn optimal trajectories communicating a fraction of the information. Additionally, we discuss what effects (desired and undesired) these event-based approaches have on the learning processes studied, and how they can be applied to more complex multi-agent systems.**

***Index Terms*—Event-Triggered Control, Reinforcement Learning, Distributed Systems**

I. Introduction

Over the past couple of decades, the interest in Reinforcement Learning (RL) techniques as a solution to all kinds of stochastic problems has exploded. In most cases, such techniques are applied to reward-maximizing problems, where an actor needs to learn a (sub) optimal policy that maximizes a time-discounted reward for any initial state. Specifically, when there is no dynamical model for the system or game, RL has proven extremely effective at finding optimal value functions that enable the construction of policies [1], [2] maximizing the expected reward over a time horizon. This has been done with convergence guarantees for different value function forms, one of the most common ones being Q-Learning [3], [4] or recently deep Q-Learning [5]–[8].

When the problems considered have a multi-agent nature, multi-agent theory can be combined with RL techniques as Q-Learning [9], [10]. In problems where a set of agents needs to optimize a (possibly shared) cost function through a model-free approach, this has been addressed in the form of Distributed Q-Learning [11]–[14]. Solutions often result in learning some form of shared policy (or value function) based on the trajectories and rewards of all agents. These techniques have been applied to many forms of competitive or collaborative problems [15]–[19]. In the latter, agents are allowed to collaborate to reach higher reward solutions compared to a selfish approach [15], [20]. This opens relevant questions regarding *how* and *when* to collaborate. In model free multi-agent systems, collaboration is often defined as either sharing experiences, value functions or policies, or some form of communication including current state variables of the agents. However such collaborative learning systems often include aggressive assumptions about communication between agents (as pointed out by [21], [22]). Approaches to reduce this communication are, in the framework of federated learning [23], in RL using efficient policy gradient methods [24], limiting the amount of agents (or information) that communicate [21] or allowing agents to learn how to communicate [25], [26]. These approaches focus on transmitting "simplified" data or learning graph topologies for the communication network.

For the problem of when to communicate over a given network, one can take inspiration from control theory approaches. When dealing with networks of sensors and actuators stabilizing a system, event-triggered control (ETC) allows sensor and actuator to estimate, through trigger functions, when is it necessary to communicate state samples or update controllers [27]–[29]. These concepts have been applied for efficient distributed stochastic algorithms [30], or to learn parameters of linear [31], [32] and non-linear [33], [34]. In multi-agent settings, they have also been investigated for distributed control of linear [35] and non-linear [36] systems, to reduce the number of interactions between agents [37], or to speed up distributed policy gradient methods [38], [39].

A. Main Contribution

Drawing a parallelism with a networked system, in this work we take inspiration from ETC techniques and turn the communication of a distributed Q-Learning problem [12] event-based, with the goal of reducing communication events, data storage and learning steps. The difficulty of such problem is that allowing agents to independently decide when to transmit samples may bias the resulting probability distribution of the collected data. The main contribution is then

split twofold. First, we design fully distributed trigger functions that incorporate trajectory memory which agents use to decide when to communicate samples in a distributed Q learning system. Second, we provide convergence guarantees of the resulting learning algorithm to the optimal Q function fixed point, and show how these guarantees depend (and do not depend) on the design parameters of the system. To the best of our knowledge, such ideas have not been applied before to distributed Q-Learning with convergence guarantees and formal bounds on optimality. To verify the theoretical results, we analyse experimentally how such event-based techniques result in more efficient learning in a distributed robotic path planning problem.

II. Preliminaries

We use calligraphic letters for sets and regular letters for functions $f : \mathbb{R}^m \to \mathbb{R}^n$. A function $f : \mathbb{R}_+ \to \mathbb{R}_+$ is in class $\mathcal{K}_\infty$ if it is continuous, monotonically increasing and $f(0) = 0,\ \lim_{a\to\infty} f(u) = \infty$. We use $\mathcal{F}$ as the measurable algebra (set) of events in a probability space, and P as a probability function $P : \mathcal{F} \to [0, 1]$. We use $E[\cdot]$ and $\mathrm{Var}[\cdot]$ for the expected value and the variance. We define the set of probability vectors of size n as $\mathbb{P}^n$ satisfying $p \in \mathbb{P}^n \Leftrightarrow p \geq \mathbf{0},\ \sum_{i=1}^n p_i = 1$. Similarly, we define the set of probability matrices of size $n \times n$ as $\mathbb{P}^{n\times n}$ where $\forall i,\ \sum_{j=1}^n P_{ij} = 1$. We use $\|\cdot\|_\infty$ as the sup-norm, $|\cdot|$ as the absolute value of a scalar or the cardinality of a set. A random process X_n converges to a random variable X *almost surely* (a.s.) as $n \to \infty$ if it does so with probability 1 for any event in $\mathcal{F}$.

A. MDPs and Q-Learning

We are interested in RL solutions that maximize the discounted reward sum on a MDP.

Definition 1. *[Markov Decision Process] A Markov Decision Process (MDP) is a tuple $(\mathcal{X},\mathcal{U},P,r)$ where $\mathcal{X}$ is a set of states, $\mathcal{U}$ is a set of actions, $P : \mathcal{U} \to \mathbb{P}^{|\mathcal{X}|\times|\mathcal{X}|}$ is the probability measure of the transitions between states and $r : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ is the reward for $x \in \mathcal{X}, u \in \mathcal{U}$.*

In general, $\mathcal{X},\mathcal{U}$ are finite sets. We refer to x, u as the state-action pair at time t, and x, x' as two consecutive states. We write $P_{xx'}(u)$ as the probability of transitioning from x to x' when taking action u. We denote in this work a *stochastic transition* MDP as the general MDP presented in Definition 1, and a *deterministic transition* MDP as the particular case where the transition probabilities additionally satisfy $P_{xx'}(u) \in \{0, 1\}$ (in other words, transitions are deterministic for a pair (x, u)).

The main goal of an RL problem is to find an optimal policy $\pi^* : \mathcal{X} \to \mathcal{U}$ that maximizes the expectation of the temporal discounted reward $E[\sum_{t=0}^\infty \gamma^t r(x_t, u_t)\,|\pi, x_0]$ $\forall x_0 \in \mathcal{X}$ for a given discount $\gamma \in (0, 1)$. To do this, we can use Q-Learning for the agent to learn the values of specific state-action pairs. Let the value of a state x under policy π be $V^\pi(x) := r(x, \pi(x)) + \gamma \sum_{x'} P_{xx'}(\pi(x))V^\pi(x')$. The optimal value function satisfies $V^*(x) := \max_a r(x, u) + \max_a \sum_{x'} P_{xx'}(u)\gamma V^*(x')$. Now define the Q-Values of a state-action pair under policy π as $Q^\pi(x, u) := r(x, u) + \gamma \sum_s P_{xx'}(\pi(x))V^\pi(x')$. The goal of Q-Learning is to approximate the optimal $Q^*(x, u)$ values, which satisfy

$$Q^*(x, u) := r(x, u) + \sum_{x'} P_{xx'}(u)\gamma V^*(x'), \tag{1}$$

and yield the optimal policy $\pi^*(x) := \mathrm{argmax}_a\, Q^*(x, u)$ maximizing the discounted reward. For this, the Q-values are initialised to some value $Q_0(x, u) \in \mathbb{R}\ \forall x, u$, and are updated after each transition observation $x \to x'$ with some learning rate $\alpha_t \in (0, 1)$ as

$$Q_{t+1}(x, u) = Q_t(x, u) + \alpha_t \Delta_t(\mu), \tag{2}$$

for a given sample $\mu = (x, u, r(x, u), x')$ and $\Delta_t(\mu) := r(x, u) + \gamma \max_{u'} Q_t(x', u') - Q_t(x, u)$ is the temporal difference (TD) error. The subscript t represents the number of iterations in (2). In practice, the coefficients α_t depend on each (x, u). For ease of notation we omit this dependence, and write $\alpha_t \equiv \alpha_t(x, u)$. The iteration on (2) is known to converge to the optimal Q^* function under conditions on reward boundedness and sum convergence for the rates α_t [3].

B. Distributed Q-Learning

Let us now consider the case where N agents (actors) perform exploration on parallel instances of the same MDP, with a central learner entity, generalized as a MDP with $(\mathcal{X}^N,\mathcal{U}^N, P, r)$. Then, a *distributed* Q-Learning system optimizes the discounted reward sum on such MDP. The goal of the distributed nature is to speed up exploration, and ultimately find the optimal policy π^* faster. Such a system may have different architectures regarding the amount of learner entities, parameter sharing between them, *etc*. We consider here a simple architecture where N actors gather experiences of the form $\mu_i = (x, u, r(x, u), x')_i$ following (possibly different) policies π_i. These actors send the experiences to a single central learner, where these are sampled in batches to perform gradient descent steps on a single $\hat{Q}$ estimator, and updates each agent's policy π_i if needed. This is a typical architecture on distributed Q-Learning problems where exploring is much less computationally expensive than learning [12].

Definition 2. *A* distributed Q-learning *system (d-Q) for a MDP $(\mathcal{X}^N,\mathcal{U}^N, P, r)$ is a set of actor agents $\mathcal{N} = \{1, 2, ..., N\}$ exploring transitions and initialised at the same $x_0 \in \mathcal{X}$, together with a single central learner agent storing a $\hat{Q} : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ estimator function. Let the subsets $\mathcal{N}_x = \{i \in \mathcal{N} : x_i = s\}$ have cardinality N_x. The estimator function is updated with samples $\mu_i\ \forall i \in \mathcal{N}$ as:*

$$\hat{Q}_{t+1}(x, u) = \hat{Q}_t(x, u) + \alpha_t \frac{1}{N_x} \sum_{i\in\mathcal{N}_x} \hat{\Delta}_t(\mu_i). \tag{3}$$

We consider the following Assumption to ensure persistent exploration.

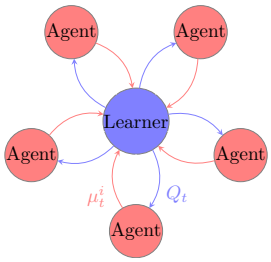

Fig. 1: Distributed Q-Learning System

Assumption 1. *Any policy π_i $\forall i \in \mathcal{N}$ has a minimum probability ε of picking an action at random.*

It is straight-forward to show that the distributed form of the Q-Learning algorithm in (3) converges to the optimal Q^* with probability one under the same assumptions as in Theorem 1 in [3]. An example architecture of a distributed Q-Learning system is shown in Figure 1.

III. Problem Definition

In practice, the distributed system in (3) implicitly assumes that actors provide their experiences at every step to the central learner that performs the iterations on the estimator $\hat{Q}$. When using large amounts of actors and exploring MDP's with large state-spaces, this can result in memory and data transmission rate requirements that scale badly with the number of actors, and can become un-manageable in size. Additionally, when the MDP has a unique initial state (or a small set thereof), the memory may become saturated with data samples that over-represent the regions of the state-space close to x_0. From this framework, we present the problem addressed in this work.

Problem 1. *For a distributed Q-learning system, design logic rules for the agents to decide when to communicate information to a central learner (and when not to) that maintain convergence guarantees and reduce the system's communication requirements.*

We consider the communication to happen from explorers to learner and from learner to explorers (star topology), and a communication event is an agent sending a sample μ_i to the central learner.

IV. Efficient Distributed Q-Learning

From the convergence proofs of Q-Learning, we know $\lim_{t\to\infty} \hat{Q}_t(x,u) - Q^*(x,u) = 0$ *a.s.* Each explorer agent obtains samples $\mu_i = (x, u, r(x,u), x')_i$, and has an estimator function $\hat{Q}_t$. For every sample, the agent can compute the estimated loss with respect to the estimator $\hat{Q}_t$, which is an indication of how far the estimator is from the optimal Q^*. This suggests that, for $\beta \in (0,1)$, we can define the surrogate function for convergence certification to be a TD error tracker:

$$L_i(t+1) := (1-\beta)L_i(t) + \beta|\hat{\Delta}_t(\mu_i)|, \forall i \in \mathcal{N}, \quad (4)$$

with $L_i(0) = 0$. We could now use L to trigger communications analogously to the role of Lyapunov functions in ETC. The parameter β serves as a temporal discount factor, that helps the agent track the TD error smoothly. Observe that $L_i(t) \geq 0\ \forall t, i$, and $L_i(t) \to 0 \Rightarrow \hat{Q}_t - Q^* \to 0$. This last property is an equivalence only in the case that the MDP has deterministic transitions. The intuition about this surrogate function is as follows. Agents compute the error term $\hat{\Delta}_t(\mu_i)$ as they move through a trajectory, which gives an indication of how close their $\hat{Q}_t$ estimator is to the optimal Q^*. Then, they accumulate these losses in a temporal discounted sum $L_i(t)$, such that by storing only one scalar value they can estimate the cumulative loss in the recent past.

Recall that in (1) the optimal Q^* represents the maximum expected Q values at every time step. Our convergence surrogate function in (4) computes the norm of the TD error at each time step, therefore it may not go to zero for stochastic transitions, but to a neighbourhood of zero.

Proposition 1. *Consider a distributed Q-Learning problem from Definition 2. For a deterministic MDP, $\hat{Q}_t \to Q^*$ a.s. $\Leftrightarrow$ $L_i(t) \to 0$ a.s. For a stochastic MDP, $\hat{Q}_t \to Q^*$ a.s. $\Rightarrow$ $L_i(t) \to \mathcal{L}_0$ a.s., where $\mathcal{L}_0 = [0, l^*]$, and $l^* = \gamma \max_{x,u,x'} \left(E[\max_{u'} Q^*(x',u')|x,u] - \max_{u'} Q^*(x',u')\right)$.*

Proposition 1. Consider first a deterministic MDP. In this case, $P_{xx'}(u) \in \{0,1\}$, and $Q^*(x,u) = r(x,u) + \gamma \max_{u'} Q^*(x',u')$. Then, it must hold

$$\hat{Q}_t \to Q^* \ a.s. \Leftrightarrow \hat{Q}_t(x,u) - Q^*(x,u) \to 0 \ a.s. \ \ \forall x,u.$$

Recalling the $Q-$learning iteration, $\forall (x,u)$ it holds *a.s.*:

$$\lim_{t\to\infty} \hat{Q}_t(x,u) = Q^*(x,u) \Leftrightarrow \lim_{t\to\infty} \hat{Q}_{t+1}(x,u) - \hat{Q}_t(x,u) =$$
$$= 0 \Leftrightarrow \lim_{t\to\infty} \hat{Q}_t(x,u) + \alpha_t \frac{1}{N_x} \sum_{i\in\mathcal{N}_x} \hat{\Delta}_t(\mu_i) -$$
$$- \hat{Q}_t(x,u) = 0 \Leftrightarrow \lim_{t\to\infty} \frac{1}{N_x} \sum_{i\in\mathcal{N}_x} \hat{\Delta}_t(\mu_i) = 0. \quad (5)$$

At last, in a deterministic MDP for a fixed (x,u) we have $\hat{\Delta}_t(\mu_i) = f(x,u)\, \forall i \in \mathcal{N}_x$. Therefore, $\frac{1}{N_x}\sum_{i\in\mathcal{N}_x} \hat{\Delta}_t(\mu_i) \to 0 \Leftrightarrow |\hat{\Delta}_t(\mu_i)| \to 0$, and $\lim_{t\to\infty} |\hat{\Delta}_t(\mu_i)| = 0 \Leftrightarrow \lim_{t\to\infty} L(t) = 0$, which happens almost surely. For the stochastic transition MDP, the optimal Q^* function satisfies

$$Q^*(x,u) = r(x,u) + \sum_{x'} P_{xx'}(u) \gamma \max_{u'} Q^*(x',u').$$

Therefore, $\lim_{t\to\infty} \hat{Q}_t(x,u) = Q^*(x,u)$ and for any $i \in \mathcal{N}_x$:

$$\lim_{t\to\infty} \hat{\Delta}_t(\mu_i) = \lim_{t\to\infty} r(x_i,u_i) + \gamma \max_{u_i'} \hat{Q}_t(x_i',u_i') -$$
$$- \hat{Q}_t(x_i,u_i) = \gamma \max_{u'} Q^*(x_i',u') -$$
$$- \gamma \sum_{x'} P_{x_i x'}(u_i) \max_{u'} Q^*(x',u') =: \hat{\Delta}^*(\mu_i) \ a.s.$$

At last, from Assumption 1, all (x,u) are visited infinitely often. Therefore, $\|\hat{\Delta}^*(\mu)\|_\infty = \gamma \|E[\max_{u'} Q^*(x',u')|x,u] -$

$\max_{u'} Q^*(x',u')\|_\infty := l^*$, and $L(t+1) \leq (1-\beta)L(t) + \beta l^* \Rightarrow \lim_{t\to\infty} L(t) \leq l^*$ *a.s.* □

A. Event Based Communication

Just as in decentralised ETC [28] a surrogate for stability (Lyapunov function) can be employed to guide the design of communication triggers, we propose here using the distributed signals $L_i(t)$ based on the TD error of the Q estimators. In our problem's context, the actors can be considered to be the sensors/actuators, and the central controller computes the iterations on $\hat{Q}_t$ based on the samples sent by the actors. This central controller updates everyone's control action (policy π_i). In the d-Q system, the state variable to stabilize is the difference $E[\hat{Q}_t|\mathcal{F}_t] - Q^*$, where $\mathcal{F}_t$ is a $\sigma-$algebra composed by the sequence of explored triplets (x,u,x'). The control action analogy is $\hat{\Delta}(\mu)$, and applying the control action results in $\|E[\hat{Q}_{t+1}|\mathcal{F}_t] - Q^*\|_\infty < \|\hat{Q}_t - Q^*\|_\infty$ (see [40]). To decide when to transmit, we propose to use triggering rules of the form

$$\theta_t(i) := \begin{cases} 1 & \text{if } |\hat{\Delta}_t(\mu_i)| \geq \max\{\rho L_i(t), \epsilon\} \\ 0 & \text{else,} \end{cases} \tag{6}$$

with $\rho \in [0,1]$ and $\epsilon \geq 0$. That is, $\theta_t(i) = 1$ means that agent i sends the sample μ_i at time t to the central learner. The event triggered rule in (6) has an intuitive interpretation in the following way. Agents accumulate the value of $L_i(t)$ through their own trajectories in time. If the trajectories sample states that are already well represented in the current $\hat{Q}_t$ function, there is no need to transmit a new sample to the central learner. This can happen for a variety of reasons; some regions of the state-space may be well represented by a randomized initialisation of $\hat{Q}$, or some explorers may have already sampled the current trajectory often enough for the learner to approximate it. The resulting system is then an event-based d-Q (EBd-Q) system. We divide now the results in stochastic and determinstic MDPs.

B. Deterministic MDPs

Let us first define H to be the operator:

$$H(\hat{Q}_t(x,u)) := \sum_{x'} P_{xx'}(u)\left(r(x,u) + \gamma \max_{u'} \hat{Q}_t(x',u')\right). \tag{7}$$

The mapping H is a contraction operator on the $\infty-$norm, with $H(Q^*) = Q^*$ being the only fixed point (see [40] for the proof). Now observe, for a deterministic MDP, that the transition $(x,u) \to x'$ happens for a single x', and $E[\hat{\Delta}_t(u)\,|\,\mathcal{F}_t] = H(\hat{Q}_t(x,u)) - \hat{Q}_t(x,u) = \hat{\Delta}_t(\mu)$.

Theorem 1. *Let $(\mathcal{X}^N, \mathcal{U}^N, P, r)$ be a deterministic d-Q. Let the event triggering condition determining communication events be* (6)*. Then, the resulting* EBd-Q *system learning on the samples $\{\mu_i : \theta_t(i) = 1\}$ converges* a.s. *to a $\hat{Q}_\epsilon$ satisfying $\|\hat{Q}_\epsilon - Q^*\|_\infty \leq f(\epsilon)$ with $f(\epsilon) \in \mathcal{K}_\infty$ under the same conditions as in [3].*

Proof. We show this by contradiction. Assume first that $\exists t_0$ such that a communication event is never triggered for $t > t_0$. Then, $|\hat{\Delta}_t(\mu_i)| < \max\{\rho L_i(t), \epsilon\}$. Take first $\max\{\rho L_i(t),\epsilon\} = \epsilon \Rightarrow |\hat{\Delta}_t(\mu_i)| < \epsilon$. Now take $\max\{\rho L_i(t),\epsilon\} = \rho L_i(t) \Rightarrow |\hat{\Delta}_t(\mu_i)| < \rho L_i(t)$, and observe $L_i(t+1) \leq (1-\beta(1-\rho))L_i(t)$. Therefore, $\exists t_\epsilon \geq t_0 : L_i(t_\epsilon) < \epsilon \Rightarrow |\hat{\Delta}_{t_\epsilon}(\mu_i)| < \epsilon$, and from (1) $\forall (x,u), t > t_\epsilon$:

$$\begin{aligned}
&|r(x,u) + \gamma \max_{u'} \hat{Q}_t(x',u') - \hat{Q}_t(x,u)| \leq \epsilon \Rightarrow \\
\Rightarrow &|Q^*(x,u) - \hat{Q}_t(x,u) + \\
&+\gamma\left(\max_{u'} \hat{Q}_t(x',u') - \max_{u'} Q^*(x',u')\right)| \leq \epsilon \Rightarrow \\
\Rightarrow &|Q^*(x,u) - \hat{Q}_t(x,u)| \leq \epsilon + \\
&+\gamma \max_{u'} \left|\hat{Q}_t(x',u') - Q^*(x',u')\right| \leq \epsilon + \gamma\|\hat{Q}_t - Q^*\|_\infty \Rightarrow \\
\Rightarrow &\|Q^* - \hat{Q}_t\|_\infty \leq \epsilon + \gamma\|\hat{Q}_t - Q^*\|_\infty \leq \frac{\epsilon}{1-\gamma},
\end{aligned} \tag{8}$$

where $f(\epsilon) := \frac{\epsilon}{1-\gamma}$ is a $\mathcal{K}_\infty$ function. Furthermore, it follows from (6) that no samples are transmitted for $t > t_\epsilon$, therefore $\hat{Q}_t$ has converged for $t > t_\epsilon$ to some $\hat{Q}_\epsilon$. Therefore, $\lim_{t\to\infty} \|Q^* - \hat{Q}_t\|_\infty = \|Q^* - \hat{Q}_\epsilon\|_\infty \leq \frac{\epsilon}{1-\gamma}$. Now assume that communication events happen infinitely often after some t_0. Since all pairs (x,u) are visited infinitely often, $\hat{\Delta}_{t+1}(\mu) = H(\hat{Q}_{t+1})(x,u) - \hat{Q}_{t+1}(x,u)$ and

$$\begin{aligned}
&\|\hat{\Delta}_{t+1}(\mu)\|_\infty \leq \|H(\hat{Q}_t)(x,u) - \hat{Q}_t(x,u) + \\
&+\alpha_t\left(\gamma \max_{\mu'} \hat{\Delta}_t(\mu') - \hat{\Delta}_t(\mu)\right)\|_\infty \leq \\
&= \|(1-\alpha_t)\hat{\Delta}_t(\mu) + \alpha_t\gamma \max_{\mu'} \hat{\Delta}_t(\mu')\|_\infty \leq \\
&\leq (1-\alpha_t(1-\gamma))\|\hat{\Delta}_t(\mu)\|_\infty.
\end{aligned}$$

Therefore, $\lim_{t\to\infty} \|\hat{\Delta}_t(\mu)\|_\infty = 0$, which implies no samples are transmitted as $t \to \infty$ and contradicts the *infinitely often* assumption. From (8), $\lim_{t\to\infty} \|Q^* - Q_t\|_\infty \leq \frac{\epsilon}{1-\gamma}$. □

Remark 1. *Observe that for $\epsilon = 0$ the triggering rule in 6 may result in regular (almost periodic) communications as $t \to \infty$. Setting $\epsilon > 0$ implies the number of expected communication events goes to 0 as $t \to \infty$, at the expense of $\hat{Q}_t$ converging to a neighbourhood of Q^*.*

One can show that, in the case of a deterministic MDP, convergence is also guaranteed for the case where $\alpha_t = \alpha$ is fixed. In practice, we can consider this to be the case when applying ET rules on a deterministic MDP.

C. Stochastic MDP

We now present similar results to Theorem 1 for general stochastic transition MDPs. Consider a distributed MDP as in Definition 2. Let H_P be the operator $H(\hat{Q}_t(x,u))$ as a function of the probability transition function P. Let $\mathcal{P}$ be the set of all possible transition functions for a given set of actions and states, *i.e.* $\mathcal{P} \equiv \left(\mathbb{P}^{|\mathcal{X}|\times|\mathcal{X}|}\right)^{|\mathcal{U}|}$. Define $\mathcal{T} := 2^{|\mathcal{X}|\times|\mathcal{U}|\times|\mathcal{X}|}$ as the power set of transitions for the given states and actions $\mathcal{X},\mathcal{U}$. Let $G : \mathcal{T}\times\mathcal{P} \to \mathcal{P}$ be a mapping such that given a set of transitions $\tau \in \mathcal{T}$ and a transition

function P sets the probability of all transitions τ to zero and normalizes the resulting function $G(\tau, P) = \hat{P}^\tau \in \mathcal{P}$.

Given the MDP probability measure P, we define the set $\mathcal{P}_P \subset \mathcal{P}$ as the set containing all transition functions $\hat{P}$ resulting from "deleting" any combination of transitions in P. That is, $\mathcal{P}_P := \{\hat{P}^\tau\}_{\tau \in \mathcal{T}}$. Consider now the case where we apply an event triggered rule to transmit samples on a stochastic MDP. For any pair (x, u), agent i and time t, the samples are transmitted (and learned) if $\theta_t(i) = 1$. This means that, in general, it may happen that for the set of resulting states $\mathcal{X}'(x,u) := \{x' \in \mathcal{X} : P_{xx'}(u) > 0\}$, some transitions will not be transmitted. In practice this is equivalent to applying different transition functions $\hat{P}^\tau \in \mathcal{P}_P$ at every step t. This leads to the next assumption.

Assumption 2. *There exists probability measure $v : \mathcal{P}_P \to [0,1]$ (or $v \in \mathbb{P}^{|\mathcal{P}_P|}$) that is only a function of the MDP $(\mathcal{X}^N, \mathcal{U}^N, P, r)$, the initial conditions x_0, $\hat{Q}_0$ and the parameters $\gamma, \varepsilon, \rho, \epsilon$, such that $v_{\hat{P}^\tau}$ is the probability of applying function $\hat{P}^\tau$ at any time step.*

Remark 2. *In fact it follows from the Definition of $\mathcal{P}_P$ that the dependence on ρ, ϵ must exist, given that $\rho, \epsilon = 0 \Rightarrow v_P = 1$: in this case all samples are always transmitted. In a similar way, it also holds that $\lim_{\epsilon \to \infty} v_{\hat{P}^\tau} = 0 \ \forall \hat{P}^\tau \in \mathcal{P}_P$ since in such case no samples are ever transmitted.*

Let us reflect on the implications of Assumption 2. When applying an event triggered rule in (6) to transmit samples, it may result on experiences not being transmitted if the trigger condition is not met. In practice, this can be modelled by considering different transition functions $\hat{P} \in \mathcal{P}_P$ (which have some values $\hat{P}^\tau_{xx'}(u) = 0$ compared to the original function P) applied at every time-step by every agent. Assumption 2 implies that, even though every agent uses different functions at every time-step, the probability of using each $\hat{P}^\tau \in \mathcal{P}_P$ is measurable for fixed initial conditions.

Remark 3. *Assumption 2 is necessary to obtain the convergence guarantees presented in the following results. Based on the experimental results obtained, and on the fact that agents follow on-policy trajectories which are (on average) similar for the same exploration rate ε, the Assumption seems to hold in the cases explored. However, we leave this as a conjecture, with the possibility that the assumption could be relaxed to a time-varying distribution over $\mathcal{P}_P$.*

Let us now define the operator $\tilde{H}$ as $\tilde{H}(\hat{Q}_t(x,u)) := \sum_{\hat{P} \in \mathcal{P}_P} v_{\hat{P}} H_{\hat{P}}(\hat{Q}_t(x,u))$, and let

$$\Phi_t(x,u) := \frac{1}{N_x} \sum_{i \in \mathcal{N}_x} r(x,u) + \gamma \max_{u'} \hat{Q}_t(x'_i, u'). \tag{9}$$

For a transition function P, set $\mathcal{P}_P$, and density v, define $\tilde{P}(x,u) := \sum_{\hat{P}} v_{\hat{P}} \hat{P}_{xx'}(u)$. We derive the following results.

Lemma 1. *For a given agent i and time t transmitting samples according to the triggering condition* (6)*, it holds that $E[\Phi_t(x,u) \,|\, \mathcal{F}_t] = \tilde{H}(\hat{Q}_t(x,u))$, and the operator has a fixed point $\tilde{H}(\tilde{Q}) = \tilde{Q}$ satisfying $\tilde{Q}(x,u) := \sum_{x'} \tilde{P}_{xx'}(u) \left(r(x,u) + \gamma \max_{u'} \tilde{Q}(x', u') \right)$.*

Proof. First, from Assumption 2, for a given $\hat{P}$, $E[\Phi_t(x,u) \,|\, \mathcal{F}_t, \hat{P}] = H_{\hat{P}}(\hat{Q}_t)(x,u)$. Now, by the law of total expectation and making use of $\Pr[\hat{P}] = v_{\hat{P}}$, it follows that $E[\Phi_t(x,u) \,|\, \mathcal{F}_t] = \sum_{\hat{P} \in \mathcal{P}_P} v_{\hat{P}} H_{\hat{P}}(\hat{Q}_t)(x,u)$. At last, to show that $\tilde{Q}$ is a fixed point, observe we can write

$$\begin{aligned}
&\sum_{\hat{P} \in \mathcal{P}_P} v_{\hat{P}} \sum_{x'} \hat{P}_{xx'}(u) \left(r(x,u) + \gamma \max_{u'} \tilde{Q}(x', u') \right) = \\
&= \sum_{x'} \left(\sum_{\hat{P}} v_{\hat{P}} \hat{P}_{xx'}(u) \right) \left(r(x,u) + \gamma \max_{u'} \tilde{Q}(x', u') \right) = \\
&= \sum_{x'} \tilde{P}_{xx'}(u) \left(r(x,u) + \gamma \max_{u'} \tilde{Q}(x', u') \right).
\end{aligned}$$

□

Therefore, applying $\tilde{H}(\tilde{Q})$ is equivalent to applying the contractive operator in (7) with transition function $\tilde{P}$.

Theorem 2. *Consider a d-Q system as in Definition 2. Let the event triggering condition determining communication events be* (6)*. Then, the resulting* EBd-Q *system learning on the samples $\{\mu_i : \theta_t(i) = 1\}$ converges* a.s. *to a fixed point $\tilde{Q}$ under the same conditions as in [3].*

Proof. Define $\xi_t(x,u) := \hat{Q}_t(x,u) - \tilde{Q}(x,u)$. Then, the iteration (2) applied at every time step is $\xi_{t+1}(x,u) = (1-\alpha_t)\xi_t(x,u) + \alpha_t(\Phi_t(x,u) - \tilde{Q}(x,u))$. Now, from Lemma 1,

$$\begin{aligned}
&\|E[\Phi_{t+1}(x,u) - \tilde{Q}(x,u) | \mathcal{F}_t]\|_\infty = \\
&= \|\tilde{H}(\hat{Q}_{t+1})(x,u) - \tilde{H}(\tilde{Q})(x,u)\|_\infty = \\
&= \gamma \|\tilde{P}_{xx'}(u)(\max_{u'} \hat{Q}_t(x', u') - \max_{u'} \tilde{Q}(x', u'))\|_\infty \leq \\
&\leq \gamma \|\tilde{P}_{xx'}(u)\|_\infty \|\hat{Q}_t - \tilde{Q}\|\|_\infty \leq \gamma \|\xi_t(x,u)\|\|_\infty.
\end{aligned}$$

Therefore, the expected value of the operator $\tilde{H}$ is a γ-contraction in the sup-norm, with fixed point $\tilde{Q}$, and it follows that $\|\xi_t(x,u)\|_\infty \to 0$ *a.s.* □

From Remark 2 we know that $\rho, \epsilon = 0 \Rightarrow v(P) = 1 \Rightarrow \tilde{Q} = Q^*$. Additionally, for $\hat{P}$ being the probability transition function applied at time t, it holds that $E[\hat{P}] = \tilde{P}$. But we can say something more about how the difference $P - \tilde{P}$ influences the distance between the fixed points $\|Q^* - \tilde{Q}\|_\infty$.

Corollary 1. *Let a distributed MDP with an event triggered condition as defined in* (6)*. For a given transition function P, a set of functions $\mathcal{P}_P$ and density v, $\exists c \geq 0 : \|Q^* - \tilde{Q}\|_\infty \leq c \frac{\gamma}{1-\gamma} \|P - \tilde{P}\|_\infty$.*

Proof. Recall $\tilde{H}(\tilde{Q}) = \tilde{Q}$ and $H(Q^*) = Q^*$. Then,

$$\begin{aligned}
&\|Q^* - \tilde{Q}\|_\infty = \|H(Q^*) - \tilde{H}(\tilde{Q})\|_\infty = \\
=&\|\sum_{x'} P_{xx'}(u)\left(r(x,u) + \gamma \max_{u'} Q^*(x',u')\right) - \\
&- \tilde{P}_{xx'}(u)\left(r(x,u) + \gamma \max_{u'} \tilde{Q}(x',u')\right)\|_\infty = \\
=&\gamma\|\sum_{x'} P_{xx'}(u) \max_{u'} Q^*(x',u') - \tilde{P}_{xx'}(u) \max_{u'} \tilde{Q}(x',u')\|_\infty.
\end{aligned} \tag{10}$$

Define $\hat{\Delta}P_{xx'}(u) := \tilde{P}_{xx'}(u) - P_{xx'}(u)$ and substitute in (10):

$$\begin{aligned}
&\|Q^* - \tilde{Q}\|_\infty = \gamma\|\sum_{x'} P_{xx'}(u)\left(\max_{u'} Q^*(x',u') - \right. \\
&\left. - \max_{u'} \tilde{Q}(x',u')\right) - \hat{\Delta}P_{xx'}(u) \max_{u'} \tilde{Q}(x',u')\|_\infty \leq \\
\leq&\gamma\|\sum_{x'} P_{xx'}(u) \max_{u'} |Q^*(x',u') - \tilde{Q}(x',u')|\|_\infty + \\
&+ \gamma\|\hat{\Delta}P_{xx'}(u) \max_{u'} \tilde{Q}(x',u')\|_\infty.
\end{aligned} \tag{11}$$

At last, observe $\|\sum_{x'} P_{xx'}(u) \max_{u'} |Q^*(x',u') - \tilde{Q}(x',u')|\|_\infty \leq \gamma\|Q^* - \tilde{Q}\|_\infty$. Additionally, since the reward functions are bounded, for a discount rate $\gamma \in (0,1)$ the values of $\tilde{Q}(x,u) \leq c$ are also bounded for some constant $c \in \mathbb{R}_+$. Therefore,

$$\begin{aligned}
\|Q^* - \tilde{Q}\|_\infty \leq&\gamma\|Q^* - \tilde{Q}\|_\infty + \gamma\|\hat{\Delta}P_{xx'}(u)\|_\infty\|\tilde{Q}\|_\infty \leq \\
\leq&\gamma\|Q^* - \tilde{Q}\|_\infty + c\gamma\|\hat{\Delta}P_{xx'}(u)\|_\infty \Rightarrow \\
\Rightarrow \|Q^* - \tilde{Q}\|_\infty \leq&c\frac{\gamma}{1-\gamma}\|P - \tilde{P}\|_\infty.
\end{aligned} \tag{12}$$

□

In fact, the distance $\|P - \tilde{P}\|_\infty$ is explicitly related to the probability measure v, since v determines how far $\tilde{P}$ is from the original P based on the influence of every function in the set $\mathcal{P}_P$. One can show that $\|P - \tilde{P}\|_\infty \leq (1 - v_P)|\mathcal{P}_P|$, and $1 - v_P$ is a measure of how often we use transition functions different to P, which depends on the aggressiveness of the parameters ρ, ϵ. We continue now to study experimentally the behaviour of the Event Based d-Q systems in Theorem 1 and 2 regarding the communication rates and performance of the policies obtained for a given path planning MDP problem.

V. Experiments

To demonstrate the effectiveness of the different triggering functions and how they affect the learning of Q-values over a MDP, we use a benchmark problem consisting of a path planning problem. Details on the experimental framework are found in Appendix B. The average reward and communication results for a stochastic and deterministic MDP are presented in Figure 2. We use as a benchmark a "vanilla" Distributed Q-Learning algorithm where all agents are communicating samples continuously, and we compare with different combinations of parameters for the presented *EBd-Q* systems. Comparing with other available research

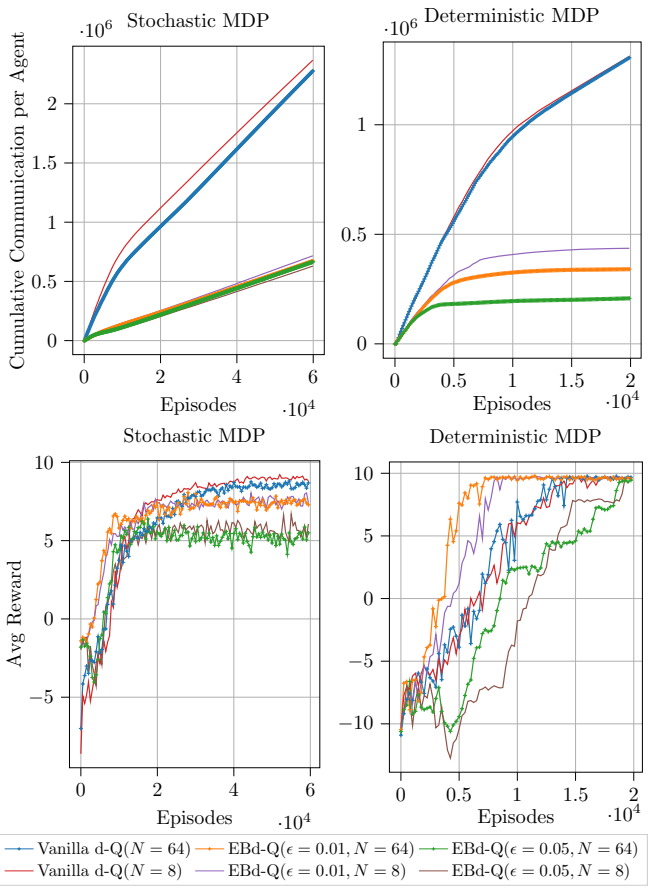

Fig. 2: Path Planning Learning, Vanilla vs. EBd-Q

is not straight-forward, since it would require interpreting similar methods designed for other problems (in the case of distributed stochastic gradient descent works [30], or policy gradient examples [38], [39]), or comparing with other methods designed for learning speed (e.g. [14]), where the goal is not to save bandwidth or storage capacity.

Analysing the experimental results, in both the stochastic and deterministic MDP scenarios, the systems reach an optimal policy quicker by following an event triggered sample communication strategy, but only for $\epsilon = 0.01$. This can be explained by the same principle as in *prioritized sampling* [13], [41]: samples of un-explored regions of the environment are transmitted (and learned) earlier and more often. However, in our case this emerges as a consequence of the trigger functions $\theta_i(t)$, and it is the result of a fully distributed decision process where agents decide independently of eachother when to share information, and does not require to accumulate and sort the experiences in the first place.

When increasing the triggering threshold to $\epsilon = 0.05$, the learning gets compromised and the reward decreases for both $N = 64$ and $N = 8$. Additionally, we observe in both scenarios how the total number of communications increase much slower in the event based case compared to the vanilla d-Q example, and even stabilize in the case of

the deterministic MDP, indicating the number of events is approaching zero. This is due to the *EBd-Q* systems sending a much lower amount of samples through the network per time step. As anticipated by the theoretical results in Theorems 1 and 2, higher ϵ results in a larger reduction of communication rates, at the expense of obtaining less optimal Q functions.

At last, let us comment on the influence of the number of agents in the experiments. First, $N = 8$ has lower communication requirements observable in the case with $\epsilon = 0.01$: the total communication number plateaus earlier and at a significantly lower value for $N = 64$. Second, in the deterministic MDP case we see how larger number of agents result in faster reaching of a maximum reward.

VI. Discussion

We have presented a design of ETC inspired trigger functions for d-Q systems with the goal to allow agents in a such systems to make distributed decisions on which particular experiences may be valuable and which ones not, reducing the amount of communication events (and data transmission and storage). Regarding the convergence guarantees, we have shown how applying such triggering functions on the communication events results in the centralised learner converging to a Q-Function that may slightly deviate from the optimal Q^*. However, we were able to provide an indication on how far the resulting Q functions can be from Q^* based on the triggering parameters ϵ, ρ, explicitly for a deterministic MDP and implicitly (via the distribution v) for a stochastic MDP. Event based rules reduce significantly the amount of communication required in the explored path planning problem, while keeping a reasonable learning speed, even though they intrinsically modify the probability distributions of the data. In fact, it was observed in the experiments how the proposed *EBd-Q* systems resulted collaterally in a faster learning rate than for the constant communication case (an effect similar to that in *prioritized learning*).

Finally, some questions for future work emerge from these results. It would be valuable to explore the effect of such event based communication on general multi-agent RL systems where all agents are learners, the communication graph has a complete topology, and agents could be sharing more than experiences ($Q-$values, policies...). Such study would shine light on how to design efficient collaborative multi agent systems. At last, we leave as a conjecture whether Assumption 2 always holds, left for future work, with the possibility of analysing *EBd-Q* systems as some form of alternating or interval MDP with probability distribution bounds.

Appendix

A. Experimental Framework

We modified the Frozen Lake environment in OpenAI GYM [42]. We edited the environment to have a bigger state-space (1296 (x, u) pairs), the agents get a reward $r = -1$ when choosing an action that makes them fall in a hole, and $r = 10$ when they find the goal state. Additionally, the agents get a constant reward of -0.01 every time they take an action, to reflect the fact that shorter paths are preferred. The action set is $\mathcal{U} = \{\text{up, down, left, right}\}$. For the stochastic transition case, the agents get a reward based on the pair (x, u) regardless of the end state x'. The resulting Frozen Lake environment can be seen in Figure 3. We consider a population of $N \in \{8, 64\}$ agents, all using ε greedy policies with different exploration rates (as proposed in [43]). The number of agents is chosen to be multiple of 8 (to facilitate running on parallel cores of the computer), to represent both a "large" and a "small" agent number scenario. The agents are initialised with a value $\varepsilon_i \in \{0.01, 0.2, 0.4, 0.6, 0.8.0.99\}$ chosen at random. For all the simulations we use $\alpha = 0.01$, $\gamma = 0.97$, $\beta = 0.05$ and $\rho = 0.9$. We plot results for $\epsilon \in \{0.01, 0.05\}$. The $Q-$function is initialised randomly $\hat{Q}_0(x, u) \in [-1, 1] \, \forall \, (x, u)$. The results are computed for 25 independent runs and averaged for each scenario. We present results for a stochastic and a deterministic MDP. In the stochastic case, for a given pair (x, u) there is a probability $p = 0.7$ of ending up at the corresponding state x' (e.g. moving down if the action chosen is *down*) and $\bar{p} = 0.3$ of ending at any other adjacent state.

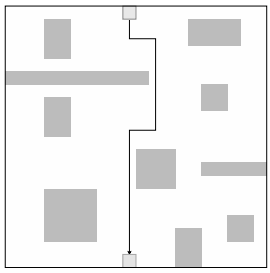

Fig. 3: Path Planning map used.

To compare between the different scenarios, we use an experience replay buffer of size $N \times 1000$ for the central learner's memory, where at every episode we sample mini-batches of 32 samples. The policies are evaluated by a critic agent with a fixed $\varepsilon_0 = 0.01$, computing the rewards for 10 independent runs for every estimation $\hat{Q}_t$. The learning rate α and "diffusion" γ were picked based on similar size Q-learning examples in the literature. In the case of the ET related parameters β, ρ, ϵ, these were picked after a very quick parameter scan. First, $\beta = 0.05$ yields a half-life time of ≈ 15 time steps, which is on the same order as the diameter of the path planning arena. The value of ρ was picked arbitrarily close to 1 to allow a slow decrease in the communication rate. At last, ϵ acts as an error threshold, under which the errors in the Q values are considered low enough and no samples are transmitted. The value function magnitude is related to the maximum reward in the MDP. Take a pair (x, u) being 1 step away from the path planning goal has an associated reward on the order of $\gamma \|r(x, u)\|_\infty \approx 9.7$. However, a pair (x, u) being 2 steps away has $\gamma^2 \|r(x, u)\|_\infty \approx 9.4$. Therefore, when being really close to the goal, the error associated with taking one extra step is on the order of ≈ 0.03. By choosing $\epsilon = 0.01$,

we ensure the threshold is low enough to capture one-step errors. Then, $\epsilon = 0.05$ is larger than this gap, so it ensures a significant enough difference for comparison.

ACKNOWLEDGEMENTS

The authors want to thank G. Delimpaltadakis, G. Gleizer and M. Suau for the useful discussions. This work is partly supported by the ERC Starting Grant SENTIENT 755953.

REFERENCES

[1] R. E. Bellman and S. E. Dreyfus, *Applied dynamic programming*. Princeton university press, 2015, vol. 2050.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[3] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[4] T. Jaakkola, M. I. Jordan, and S. P. Singh, "On the convergence of stochastic iterative dynamic programming algorithms," *Neural computation*, vol. 6, no. 6, pp. 1185–1201, 1994.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[8] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[9] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *TARK*, vol. 96. Citeseer, 1996, pp. 195–210.

[10] J. Hu, M. P. Wellman *et al.*, "Multiagent reinforcement learning: theoretical framework and an algorithm." in *ICML*, vol. 98. Citeseer, 1998, pp. 242–250.

[11] G. Weiß, "Distributed reinforcement learning," in *The Biology and technology of intelligent autonomous agents*. Springer, 1995, pp. 415–428.

[12] A. Nair, P. Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A. De Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen *et al.*, "Massively parallel methods for deep reinforcement learning," *arXiv preprint arXiv:1507.04296*, 2015.

[13] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. Van Hasselt, and D. Silver, "Distributed prioritized experience replay," *arXiv preprint arXiv:1803.00933*, 2018.

[14] S. Kapturowski, G. Ostrovski, J. Quan, R. Munos, and W. Dabney, "Recurrent experience replay in distributed reinforcement learning," in *International conference on learning representations*, 2018.

[15] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.

[16] E. Yang and D. Gu, "Multiagent reinforcement learning for multi-robot systems: A survey," tech. rep, Tech. Rep., 2004.

[17] A. Nowé, P. Vrancx, and Y.-M. De Hauwere, "Game theory and multi-agent reinforcement learning," in *Reinforcement Learning*. Springer, 2012, pp. 441–470.

[18] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[19] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv preprint arXiv:1706.02275*, 2017.

[20] M. Lauer and M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *In Proceedings of the Seventeenth International Conference on Machine Learning*. Citeseer, 2000.

[21] J. R. Kok and N. Vlassis, "Sparse cooperative q-learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 61.

[22] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous agents and multi-agent systems*, vol. 11, no. 3, pp. 387–434, 2005.

[23] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[24] T. Chen, K. Zhang, G. B. Giannakis, and T. Basar, "Communication-efficient distributed reinforcement learning," *arXiv preprint arXiv:1812.03239*, 2018.

[25] J. N. Foerster, Y. M. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *arXiv preprint arXiv:1605.06676*, 2016.

[26] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," *arXiv preprint arXiv:1912.06095*, 2019.

[27] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680–1685, 2007.

[28] M. Mazo and P. Tabuada, "Decentralized event-triggered control over wireless sensor/actuator networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2456–2461, 2011.

[29] ——, "On event-triggered and self-triggered control over sensor/actuator networks," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 435–440.

[30] J. George and P. Gurram, "Distributed stochastic gradient descent with event-triggered communication," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7169–7178.

[31] F. Solowjow and S. Trimpe, "Event-triggered learning," *Automatica*, vol. 117, p. 109009, 2020.

[32] K. G. Vamvoudakis and H. Ferraz, "Model-free event-triggered control algorithm for continuous-time linear systems with optimal performance," *Automatica*, vol. 87, pp. 412–420, 2018.

[33] X. Zhong, Z. Ni, H. He, X. Xu, and D. Zhao, "Event-triggered reinforcement learning approach for unknown nonlinear continuous-time system," in *2014 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2014, pp. 3677–3684.

[34] N. Funk, D. Baumann, V. Berenz, and S. Trimpe, "Learning event-triggered control from data through joint optimization," *IFAC Journal of Systems and Control*, vol. 16, p. 100144, 2021.

[35] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2011.

[36] K. F. Elvis Tsang and K. H. Johansson, "Distributed event-triggered learning-based control for nonlinear multi-agent systems," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 3399–3405.

[37] R. Becker, S. Zilberstein, and V. Lesser, "Decentralized markov decision processes with event-driven interactions," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*. Citeseer, 2004, pp. 302–309.

[38] Y. Lin, K. Zhang, Z. Yang, Z. Wang, T. Başar, R. Sandhu, and J. Liu, "A communication-efficient multi-agent actor-critic algorithm for distributed reinforcement learning," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 5562–5567.

[39] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[40] F. S. Melo, "Convergence of q-learning: A simple proof," *Institute Of Systems and Robotics, Tech. Rep*, pp. 1–4, 2001.

[41] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[42] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[43] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.